UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/757,517	01/10/2001	Stephen E. Fischer	FIS920000280US1	8455

29505 7590 05/19/2004

DELIO & PETERSON, LLC
121 WHITNEY AVENUE
NEW HAVEN, CT 06510

EXAMINER	
FOWLKES, ANDRE R	

ART UNIT	PAPER NUMBER
2122	

DATE MAILED: 05/19/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/757,517	FISCHER, STEPHEN E.	
	Examiner	Art Unit	
	Andre R. Fowlkes	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 4/2/04.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-7 and 9-25 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-7 and 9-25 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the replacement amendment filed on 04/02/04.

2. The objection to the drawings is withdrawn, in view of applicant's amendment.

3. The objection to the specification is withdrawn, in view of applicant's amendment.

4. The objection to claim 5 is withdrawn, in view of applicant's amendment.

5. The rejection under 35 U.S.C. 112 to claim 12 is withdrawn, in view of applicant's amendment.

6. Claims 1-11, 13-19, and 23-25 are rejected under 35 USC 102(a) as being anticipated by The GNU Make Manual, Version 3.79, edition 0.55, (04/04/2000).

7. Claims 12 and 20-23 are rejected under 35 USC 103(a) as being unpatentable over The GNU Make Manual, Version 3.79, edition 0.55, (04/04/2000) in view of Welsh, "Practical Programming in Tcl and Tk", (1999).

Claim Rejections - 35 USC § 102

8. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (a) the invention was known or used by others in this country, or patented or described in a printed publication in this or a foreign country, before the invention thereof by the applicant for a patent.

9. Claims 1-11, 14-19, and 23-25 are rejected under 35 U.S.C. 102(a) as being anticipated by The GNU Make Manual (GNU MM), Version 3.79, edition 0.55, (04/04/2000).

As per claim 1, this is another method version of the claimed method discussed below, in claim 5, wherein all claimed limitations also have been addressed below.

As per claim 2, the rejection of claim 1 is incorporated and further, the GNU Make Manual discloses **including source code and object code, said target files being source code and said dependency files being object code, said source code being selectively compiled to update said object code** (p. 6 lines 2-3, "The make utility automatically (and selectively) determines which pieces (i.e. source code) of a large program need to be recompiled (to update object code)").

As per claim 3, the rejection of claim 1 is incorporated and further, the GNU Make Manual discloses **updating said prerequisite strings with new files, defined by new code** (p. 16 lines 4-5, "a text string value ... can be substituted (i.e. updated)").

As per claim 4, the rejection of claim 2 is incorporated and further, the GNU Make Manual discloses that **said algorithm utilizes a search technique including pattern type variables which use generic rules to specify said object code for updating** (p. 6 lines 2-3, "The make utility automatically determines which pieces of a large program need to be recompiled (using rules)", and p. 42 line 17, "pattern rules", can be used).

As per claim 5, The GNU MM discloses:

- reading existing target files and existing dependency files in said computer program (p. 21 lines 41-42, "GNU make ... reads (the target files and dependency files)"),

- reading a plurality of said dependency files associated with said target files into a single control file, wherein selected lines of said dependency files are split in to target strings and prerequisite strings (p. 21 lines 41-42, "GNU make ... reads all (the files dependant on the target files)", and p. 14 line 11, "grouping entries by their prerequisites"),

- executing a utility program which updates said target files and said dependency files associated with said target file, said utility program including an interpreted-scripting language specifying particular characters to search for in said target files and said associated dependency files (p. 6 lines 2-3, "The make utility automatically determines which pieces (i.e. target files and dependency files) of a large program need to be recompiled (using rules)", and p. 47 lines 35-44, "the commands of a rule consist of shell command lines... commands in makefiles are always interpreted"),

- generating a list of target strings having interpreted scripting language substitutions (p. 9 lines 35-50, shows a list of targets ... to update, and p. 16 lines 4-5, "a text string value ... can be substituted") **and corresponding** group **for said prerequisite strings using said utility program** (p. 14 line 11, "grouping entries by their prerequisites"),

-updating said target files by employing a search technique defined in said utility program, said search technique includes specified target patterns such that said specified target patterns identify said existing target files being updated (p. 6 lines 2-3, "The make utility automatically determines which pieces of a large program need to be recompiled (using rules)", and p. 42 line 17, "(target) pattern rules", are used), **said existing target files being updated if it is determined from said specified target patterns that said prerequisite strings in said control file have been updated more recently that said substituted target string** (p. 6 lines 39-41, "The make program uses the ... last –modification times of the files to decide which of the files need to be updated").

As per claim 6, the rejection of claim 5 is incorporated and further, the GNU Make Manual discloses that **specified target patterns of said search technique includes pattern type variables which use rules to specify said target files for updating.** (p. 6 lines 2-3, "The make utility automatically determines which pieces of a large program need to be recompiled (using rules)", and p. 42 line 17, "pattern rules", are used to specify target files for updating).

As per claim 7, the rejection of claim 5 is incorporated and further, the GNU Make Manual discloses that the **search technique includes matching specified characters in said target files to said list of target strings such that said specified characters identify said existing target files being updated** (p. 6 lines 2-3, "The

make utility automatically determines which pieces (i.e. target files) of a large program need to be recompiled (using rules)", and p. 42 line 17, "pattern rules", are used).

As per claim 9, the rejection of claim 5 is incorporated and further, the GNU Make Manual discloses that **said utility program defines new target files to be added to said existing target files** (p. 6 lines 2-3, "The make utility automatically determines which pieces of a large program need to be recompiled (using rules)", and p. 42 line 17, "pattern rules", are used to define new target files to be added to existing target files).

As per claim 10, the rejection of claim 5 is incorporated and further, the GNU Make Manual discloses that **the utility program prioritizes said target files to update while employing said search technique** (The section titled "How make processes a makefile" on p. 11 lines 1-51, describes how the make utility prioritizes target files to update).

As per claim 11, the rejection of claim 5 is incorporated and further, the GNU Make Manual discloses that **said utility program includes a process procedure for an operator to call, said process procedure recursively invokes said utility program and arguments** (p. 54 lines 12-13, "Recursive use of (the) make (utility) means using make as a command in a makefile").

As per claim 14, this is a product version of the claimed method discussed above in claim 5, wherein all claimed limitations also have been addressed above, and such a product is deemed to be inherent in the GNU Make Utility, otherwise, it would be inoperative.

As per claim 15, this is a device version of the claimed method discussed above in claim 5, wherein all claimed limitations also have been addressed above, and such a product is deemed to be inherent in the GNU Make Utility, otherwise, it would be inoperative.

As per claim 16, the GNU Make Manual discloses:

- **a method for updating target files in a computer** (p. 6 lines 2-3, "The make utility automatically determines which pieces of a large program need to be recompiled, and issues commands to recompile them"),

- **generating a target file list of target files to update** (p. 9 lines 35-50, shows a list of target files to update),

- **reading into a control file a list of files dependant on said target files** (p. 21 lines 41-42, "GNU make ... reads all (the files dependant on the target files)"),

- **splitting selected lines of said dependant files into target strings and prerequisite strings** (p. 14 line 11, "grouping entries by their prerequisites"),

- **performing programming language substitutions in said target strings** (p. 16 lines 4-5, "a text string value ... can be substituted"),

- appending said substituted target strings (p. 18 lines 3, "appending (strings)"),

- storing said prerequisite strings (p. 73 line 36, "it stores the text (string)"),

-executing an algorithm to match selected target files to said substituted target string (p. 16 lines 4-5, "a text string value ... can be substituted", and for the proper substitution to be made, the strings and files must be matched),

- retrieving said prerequisite strings and updating said prerequisite strings by performing all possible programming language substitutions to said prerequisite strings using said algorithm (p. 16 lines 4-5, "a text string value ... can be substituted"),

- identifying those prerequisite strings that have been updated more recently than said substituted target string to generate update rules using said algorithm (p. 9 lines 12-14, "a rule ... explains how and when to remake certain files... make carries out the commands on the prerequisites to create or update the target"),

- updating said target files from said target file list using said update rules (p. 9 lines 12-14, "a rule ... explains how and when to remake certain files... make carries out the commands on the prerequisites to create or update the target").

As per claim 17, the rejection of claim 16 is incorporated and further, the GNU make manual discloses **after said list of files dependant on said target files are read into said control file, said remaining subsequent steps utilize said control file** (p. 21 lines 41-42, "GNU make ... reads all (the files dependant on the target files)", and p.

6 lines 2-3, "The make utility automatically determines which pieces of a large program need to be recompiled, and issues commands to recompile them").

As per claims 18 and 19, the GNU Make Manual also discloses such claimed limitations as addressed in claims 9 and 6 above, respectively.

As per claim 23, the rejection of claim 16 is incorporated and further, the GNU make manual discloses that **update rules support multi-directory builds from a single control file** (p. 28 lines 51-53, "The directory search features of make facilitate this by searching several directories automatically to find a prerequisite").

As per claim 24, the rejection of claim 16 is incorporated and further, the GNU make manual discloses that **said prerequisite uses dynamic directory switching to specify multiple files in multiple directories** (p. 33 lines 50-54, "When a prerequisite's name has (a certain form) ... make handles it by (dynamically searching in multiple directories)").

As per claim 25, the rejection of claim 16 is incorporated and further, the GNU make manual discloses **said algorithm considering said directory to be out-of-date regardless of its time stamp such that any rule associated with directory target is always triggered** (p. 120 line 5 – p. 127 line 50 , "(pattern matching rules and automatic variables are used to specify rules for directories)").

Claim Rejections - 35 USC § 103

10. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

11. Claims 12, 13, 20-22 are rejected under 35 USC 103(a) as being unpatentable over The GNU Make Manual, Version 3.79, edition 0.55, (04/04/2000) in view of Welsh, "Practical Programming in Tcl and Tk", (1999).

As per claim 12, the rejection of claim 5 is incorporated and further, the GNU Make Manual doesn't explicitly disclose that **said utility program is in a scripting language selected from the group consisting of updt, perl, and Tcl**.

However, Welch, in an analogous environment, discloses that **said utility program is in a scripting language selected from the group consisting of updt, perl, and Tcl** (p. 132 lines 8-10, "The (Tcl) subst command looks through the data for square brackets, dollar signs, and backlashes, and it does substitutions on those").

Therefore, it would have been obvious to a person of ordinary skill in the art, at the time the invention was made, to incorporate the teachings of Welch into the system of GNU Make utility to have a the utility program in a language selected from updt, perl, and Tcl. The modification would have been obvious because one of ordinary skill in the

art would have wanted to use Tcl because it is useful language that allows one to build complex programming scripts yet it is easy to combine with current applications (Welch p. xivi, ¶s 3 and 4).

As per claim 13, the rejection of claim 5 is incorporated and further, the GNU Make Manual discloses that **said utility program provides that said existing target files with a specific character are not considered to be a file, and thereby are bypassed for any changes by the utility program**.

However, Welch, in an analogous environment, discloses that **said utility program provides that said existing target files with a specific character are not considered to be a file, and thereby are bypassed for any changes by the utility program** (p. 132 lines 8-10, "The subst command looks through the data for square brackets, dollar signs, and backlashes, and it does substitutions on those. It leaves the rest alone", and the specific character used to cause subst to ignore the file is a blank space character).

Therefore, it would have been obvious to a person of ordinary skill in the art, at the time the invention was made, to incorporate the teachings of Welch into the system of the GNU Make utility to have a specific character used to instruct the system to bypass the file containing the specific character. The modification would have been obvious because one of ordinary skill in the art would want to have the flexibility of specifying if a file is to be scanned and possible modified.

As per claims 20 and 21, the GNU Make utility and Welch combination also discloses such claimed limitations as addressed in claim 12, above.

As per claim 22, the rejection of claim 21 is incorporated and further, the GNU Make manual discloses that **said rules have access to said interpreted programming language to recursively invoke said algorithm on a new target** (p. 54 lines 12-13, "Recursive use of (the) make (utility) means using make as a command in a makefile").

Response to Amendment

11. Applicant's arguments with respect to claim 5 have been considered but are moot in view of the new grounds of rejection.

Conclusion

12. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the

shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Andre R. Fowlkes whose telephone number is (703)305-8889. The examiner can normally be reached on Monday - Friday, 8:00am-4:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (703)305-4552. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

ARF

TUAN DAM
SUPERVISORY PATENT EXAMINER